

COPY

UNITED STATES PATENT APPLICATION

FOR

APPARATUS FOR TRANSMITTING AND RECEIVING EXECUTABLE  
APPLICATIONS AS FOR A MULTIMEDIA SYSTEM, AND METHOD AND  
SYSTEM TO ORDER AN ITEM USING A DISTRIBUTED COMPUTING SYSTEM

INVENTORS:

Joseph Kuriacose

Ainsley Wayne Jessup, Jr.

Vincent Dureau

Alain Delpuch

Prepared by:

BLAKELY, SOKOLOFF, TAYLOR & ZAFMAN  
12400 WILSHIRE BOULEVARD  
SEVENTH FLOOR  
LOS ANGELES, CALIFORNIA 90025  
(408) 720-8598

Attorney's Docket No. 005214.P001R

"Express Mail" mailing label number: EL471466794US

Date of Deposit: September 27, 2000

I hereby certify that I am causing this paper or fee to be deposited with the United States Postal Service "Express Mail Post Office to Addressee" service on the date indicated above and that this paper or fee has been addressed to the Assistant Commissioner for Patents, Washington, D. C. 20231

Lindy Vjaretti

(Typed or printed name of person mailing paper or fee)

Lindy Vjaretti  
(Signature of person mailing paper or fee)

09-27-00  
(Date signed)

00503091-071001

The present invention relates to a client-server distributed computer system. Such a computer system has application in broadcast multimedia applications.

Early computer systems were standalone systems, consisting generally of mainframe computers. Later, several mainframe computer systems were closely connected, or clustered, to handle larger computing jobs, such as a large number of time sharing users. With the advent of personal computers, large numbers of relatively low power standalone computer systems were controlled directly by their users. Soon these large numbers of personal computers were coupled together into networks of computers, providing shared resources and communications capabilities to the users of the individual personal computers and between those users and the preexisting mainframe computers.

One form of such a network includes a central computer, 20  
called a server, which generally includes a large amount of  
mass storage. Programs used by the network users are  
centrally stored in the mass storage on the server. When a  
user desires to run a program, the user's computer requests 25  
that a copy of that program be sent to it from the server. In  
response to that request, the server transfers a copy of the  
program from its mass storage to the main memory of the  
personal computer of that user, and the program executes on  
that personal computer. Data also may be centrally stored in 30  
the server and shared by all the users on the network. The  
data is stored on the mass storage of the server, and is  
accessible by all the network users in response to a request.  
The server also serves as a hub for communications of  
messages (electronic mail) between network users. The 35  
server in such a system handles the storage and distribution  
of the programs, data and messages, but does not contribute  
any processing power to the actual computing tasks of any  
of the users. I.e. a user cannot expect the server computer to  
perform any of the processing tasks of the program execut- 40  
ing on the personal computer. While such networks perform  
a valuable function, they are not distributed computing  
systems, in which interconnected computers cooperate to  
perform a single computing task.

In an improvement to such networks, the network may be configured in such a manner that a user on the network may request that the server, or other personal computer connected to the network, execute a program. This is termed remote execution because a computer (server or other personal computer) remote from the requester is executing a program in response to a request from the requester. In such a system, the program of which remote execution is requested is either sent from the requester to the remote computer, or retrieved from the server in response to a request by the remote computer. When the program is received, it is executed. In this manner several computers may be enlisted to cooperate in performing a computing function.

Recently, there have been programs which distribute the actual computing tasks necessary for performing a single computing function. For example, in such a data base program, where the data base is stored in the mass storage of the server, if a user desires to make a query of the data base, the portion of the data base management program on that user's personal computer will generate a query request, which is forwarded to the server. The portion of the data base management program on the server performs the query processing, e.g. parsing the query request, locating where the data specified in the query request resides on its mass

55 In all of the above systems, a central computer controls or responds to requests from peripheral computers attached to it through a network. I.e. the peripheral computer (personal computer) requests remote execution of a program, requests a file or message from, or sends a query  
60 request to, another computer. Only in response to a request does the other computer provide a response, e.g. remote execution, the requested file, message or retrieved data. In addition, in general, the peripheral computer is required to have all the resources necessary to completely, or almost  
65 completely, execute the desired program, with the server acting only as another storage mechanism or at most sharing a portion of the computing tasks.



is coupled to a plurality of client computers 20 through a transport mechanism 30. The server computer 10 may be coupled to more than the three client computers 20 illustrated in FIG. 1, and the client computers 20 may be geographically widely dispersed. Client computer 22 is bidirectionally coupled to a local computer 40, to an auxiliary data processing system 50 and to a central processing facility 60. The central processing facility 60 is bidirectionally coupled to the server computer 10. The central processing facility 60 may also be connected to facilities other than the server computer 10 illustrated in FIG. 1. The local computer 40 is further bidirectionally coupled to a mass storage device 70. The client computer 22 interacts with a user 80 by providing information to the user via a display screen or other output device (not shown) and by accepting information from the user via a keyboard or other input device (also not shown).

Client computers 24 and 26 also interact with their users, (not shown in order to simplify the drawing). In addition, client computers 24 and 26 are bidirectionally coupled to the central processing facility 60. Such links are optional, however. The only requirements for any client computer 20 is a way to interact with a user, and a connection to the transport mechanism 30. Links to local computers, auxiliary data processing systems, and the central processing facility 60 are all optional, and need not be present in every one of the client computers 20.

The transport mechanism 30 includes a unidirectional high speed digital data link, such as a direct fiber optic or digital satellite link from the server 10 to the client computers 20. The data may be transported over the transport system 30 by a packet data system. In such a system, a stream of data packets, each including identification information indicating, among other things, the type of data contained in that packet and the actual data, is transmitted through the data link. Such a packet data system allows several independent streams of data, each identified by identification information in their packets, to be time multiplexed within a single stream of packets.

In addition, it is possible to multiplex a plurality of such packet data streams over respective channels on the same physical medium (fiber optic or satellite radio link) making up the transport mechanism 30. For example, different data streams may be modulated on carrier signals having different frequencies. These modulated carriers may be transmitted via respective transponders on a satellite link, for example. Further, if a particular transponder has sufficient capacity, it is possible to time multiplex several data streams on a single modulated carrier.

The client computers 20 each contain a data receiver for selecting one of the streams of packets being transported over the transport mechanism 30, receiving the selected stream of packets and extracting the data contained in them. Continuing the above example, the data receiver may include a tunable demodulator for receiving one of the respective modulated carriers from the satellite link. In addition, the data receiver may include circuitry for time demultiplexing the respective data streams being carried by that modulated carrier.

In operation, the server 10 produces a continuous data stream in the form of a stream of packets for the client computers 20. The server 10 repetitively inserts a packet, or successive packets, containing data representing the distributed computing application, including at least one executable code module, into the data stream. This code module contains executable code for the client computers 20. The data receiver in, for example, client computer 22, continu-

Any of the client computers 20 may join the distributed computing function represented by the packet stream at any 65 time, and each of the client computers 20 may operate at its own speed, generally in response to the user 80. In order to

5

19

39

50

65

5

10

15

20

25

40

45

65



In another example, the distributed computing system may be an interactive television system, broadcasting a home shopping show as the distributed computing application. In such a case, the auxiliary data carries the video and audio portion of the television signal, which may show and describe the items being offered for sale, and may include both live actors and overlaid graphics generated at the central studio. Code and data modules making up the interactive television application may include data about the products which will be offered for sale during this show, or portion of the show, and executable code to interact with the user in the manner described below.

It is also possible that permanent information about the viewer (i.e. the name, address, method of payment and credit card number) may be preentered once by the viewer, so it is not necessary to solicit that information each time an order is placed. The information is stored in permanent memory in the client computer. In such a case, when an order is placed, that information is retrieved from the permanent memory, appended to the item number and transmitted to the central computer. It is further possible that, by means of time codes, or other commands, inserted into the data stream, the client computer will know which item is currently being offered for sale. In such a case, the viewer will be able to order it by simply pressing one button on the TV remote control. In response, the client computer can combine the previously received information related to the item currently being offered for sale with the previously stored personal information about the viewer.

In operation, data representing the distributed computing application program, and data related to the transmission of the program over the transport mechanism 30 are supplied to the flow builder 102 from the application source 101. This data may be supplied either in the form of files containing data representing the code and data modules, or by scripts providing information on how to construct the code and data modules, or other such information. The code and data modules may be constant or may change dynamically, based on inputs received from the client computers 20 via the central computing facility 60 and/or other sources. The executable code and data module files may be generated by a compiler, interpreter or assembler in a known manner in

**§**

10

25

4

6

FIG. 3 is a timing diagram illustrating the data streams produced by the server computer 10 in a distributed computing system as illustrated in FIG. 1. In FIG. 3 server computer 10 is shown as simultaneously producing a plurality of packet streams 32-38. Each packet stream (32-38) is shown as a horizontal band divided into packets having the same duration and number of bits. As described above, it is possible that the size of the packets within any packet stream vary with the amount of data to be carried. In FIG. 3 it can be seen that the starting times of the packets are not synchronized. It is possible to synchronize the packets, but it is not necessary. In FIG. 3, packets carrying data representing directories are designated DIR, packets carrying data representing code modules are designated CM, packets carrying data representing data modules are designated DM, and packets carrying auxiliary data are designated AUX.

In the top series of packets 32, the leftmost packet contains data representing a code module, CM. This is followed by three packets containing auxiliary data, AUX, followed by another packet containing data representing the code module, CM. From the series of packets 32 it can be seen that the code module is repetitively produced. There may be more or fewer packets in between successive repetitions of the code module packets CM. The rate of repetition may be specified by the programmer when the application is programmed, and may be varied during the execution of the application.

In the next series of packets 34, the leftmost packet 35 contains auxiliary data, AUX. The next two packets contain respective portions of a code module (CM1,CM2). The last packet contains auxiliary data, AUX. From the series of packets 34 it can be seen that if a code module is too large to be contained in a single packet, it may be carried by more 40 than one, with each packet containing a portion of the code module. Although two packets are illustrated in the series of packets 34 as containing the code module (CM1,CM2), any number of packets may be used to carry the code module, depending upon its size. The two packets carrying the code 45 module, (CM1,CM2) are repetitively transmitted (not shown) in the series of packets 34, as described above.

In the series of packets 36, the leftmost packet contains data representing a code module (CM). The next packet (DM1) is a first packet containing data representing a data module. The next packet contains auxiliary data, AUX. The next packet (DM2) is a second packet containing the remaining data representing the data module. From the series of packets 36 it may be seen that a data module (DM1,DM2), associated with the code module (CM), may also be included in the packet stream. Both the code module (CM) and the data module (DM1,DM2) are repetitively transmitted (not shown) in the series of packets 36. The rate of repetition of the code module (CM) may be different from that of the data module (DM1,DM2), and both rates may be specified by the application programmer and varied during the execution of the application.

It may further be seen that if the data module is too large to be contained in a single packet, it may be carried by more than one packet, with each packet containing a portion of the data module. Although two packets are illustrated in the series of packets 36 as containing the data module (DM1,

DM2), any number of packets may be used to carry the data module, depending upon its size. It may be further seen that the packets carrying the data module need not be transmitted sequentially, but may have intervening packets in the packet stream. The same is true for multiple packets carrying a code module or directory module (not shown).

In the bottommost series of packets 38, the leftmost packet contains data representing the directory (DIR). The next packet contains data representing a code module (CM), followed by a packet containing auxiliary data (AUX) and a packet containing data representing a data module (DM). In the series of packet 38 all of a directory module (DIR), a code module (CM) and a data module (DM) in a single packet stream may be seen. The respective repetition rates of these three modules may be different, as specified by the programmer of the application, and may be varied during the execution of the application.

FIG. 4 is a block diagram of a client computer 22 as illustrated in FIG. 1. In FIG. 4, transport mechanism 30 (of FIG. 1) is coupled to an input terminal of a stream selector 202. An output terminal of stream selector 202 is coupled to respective input terminals of an auxiliary data extractor 204 and a packet data extractor 206. An output terminal of auxiliary data extractor 204 is coupled to the auxiliary data processor 50 (of FIG. 1). A bidirectional terminal of packet data extractor 206 is coupled to a corresponding terminal of a stream I/O adapter 208. A control output terminal of stream I/O adapter 208 is coupled to a corresponding control input terminal of stream selector 202. The combination of stream selector 202, auxiliary data extractor 204 and packet data extractor 206 form a data stream receiver 207 for client computer 22, illustrated by a dashed line in FIG. 4.

Stream I/O adapter 208 forms a part of a processing unit 224 in client computer 22, illustrated by a dashed line in FIG. 4. In addition to the stream I/O adapter 208, processing unit 224 includes a processor 210, read/write memory (RAM) 212 and read-only memory (ROM) 214 coupled together in a known manner via a system bus 216. Further input and output facilities are provided by an I/O port 218, coupled to the local processor 40 (of FIG. 1); user I/O adapter 220, for communicating with user 80; and modem 222, coupled to the central processing facility 60 (of FIG. 1); all also coupled to the system bus 216 in a known manner. Other adapters (not shown) may be coupled to system bus 216 to provide other capabilities to the processing unit 224.

As described above, auxiliary data extractor 204, I/O port 218 and modem 222 are not required in a client computer 20 according to the present invention. They are illustrated in FIG. 1 and FIG. 4 to show optional additional functionality.

In operation, processor 210 of processing unit 224 retrieves program instructions permanently stored in ROM 214, or temporarily stored in RAM 212, and executes the retrieved instructions to read data from ROM 212 and/or RAM 214, write data to RAM 212 and/or receive data from or supply data to outside sources via the I/O port 218, user I/O adapter 220 and/or modem 222, in a known manner. Under program control, processor 210 may also request a code and/or data module from the data stream supplied to the client computer 22 via the transport mechanism 30 (of FIG. 1). To retrieve this data, processor 210 first instructs stream I/O adapter 208 to send a selection control signal to the stream selector 202, possibly in response to user input from user I/O adapter 220. Then processor 210 issues a request for a specific code or data module to the stream I/O adapter 208. Stream I/O adapter 208 relays this request to the packet data extractor 204.

Transport mechanism 30 (of FIG. 1) supplies all of the plurality of packet: streams (32-38 of ) it carries to the

An interactive TV system, may also include a sound adapter coupled to the system bus 216. The sound adapter 65 generates a signal representing a computer generated sound (such as music, synthesized voice or other sound), in a